Introduction

TDDC90 – Software Security

Ulf Kargén

Department of Computer and Information Science (IDA)

Division for Database and Information Techniques (ADIT)

Agenda

- Why study software security?
- Organization of the course
  - Course contents
  - Prerequisites
  - Lectures overview
  - Labs
  - Reading material

Course leader
Nahid Shahmehri

Course assistant
Ulf Kargén

Case study 1

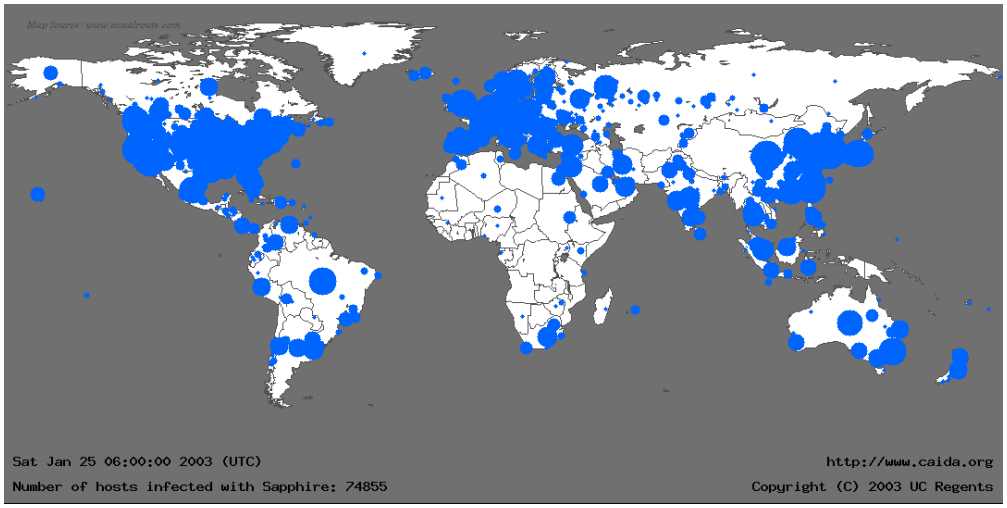
SQL Slammer

January 2003

The problem

- Stack-based buffer overflow in MS SQL server 2000
  - One UDP packet of 376 bytes let attacker run arbitrary code with privileges
  - Avg. 4000 scan attempts per second; 90% of vulnerable hosts infected in 10 minutes
  - Initial byte 0x04 causes SQL server to generate long registry key

- By supplying a carefully crafted attack packet, an adversary could take over the SQL server process



The damage

- About 75000 machines
  - Bank of America: ATMs unavailable
  - Continental Airlines: delayed and canceled flights
  - City of Seattle: 911 emergency network down
- Similar worms (2003)
  - CSX railways: traffic disruptions for one week
  - Canadian Airlines: canceled flights
  - Businesses, government shut down
- Approximate damages: Way more than $1 billion

Case study 2

Stuxnet

June 2010

The advent of ”cyber warfare”?

- Presumably designed to **physically** destroy centrifuges in an Iranian nuclear enrichment facility.
- Used four (4) previously unknown vulnerabilities (zero-days) in Windows to silently infect machines
  - Spread using (among others) infected USB sticks to reach systems not connected to the internet.
- When target system was reached
  - Reprogrammed industrial controllers to spin centrifuges out of control
  - Intercepted communication with control-room to tell operators everything was OK
- Generally believed to have been developed by US and Israeli intelligence agencies

Software security today

- 10-15 years ago: Most attacks still carried out “for fun”
- Today: Attacks almost exclusively motivated by political or economical gains (organized crime, espionage, hacktivism)

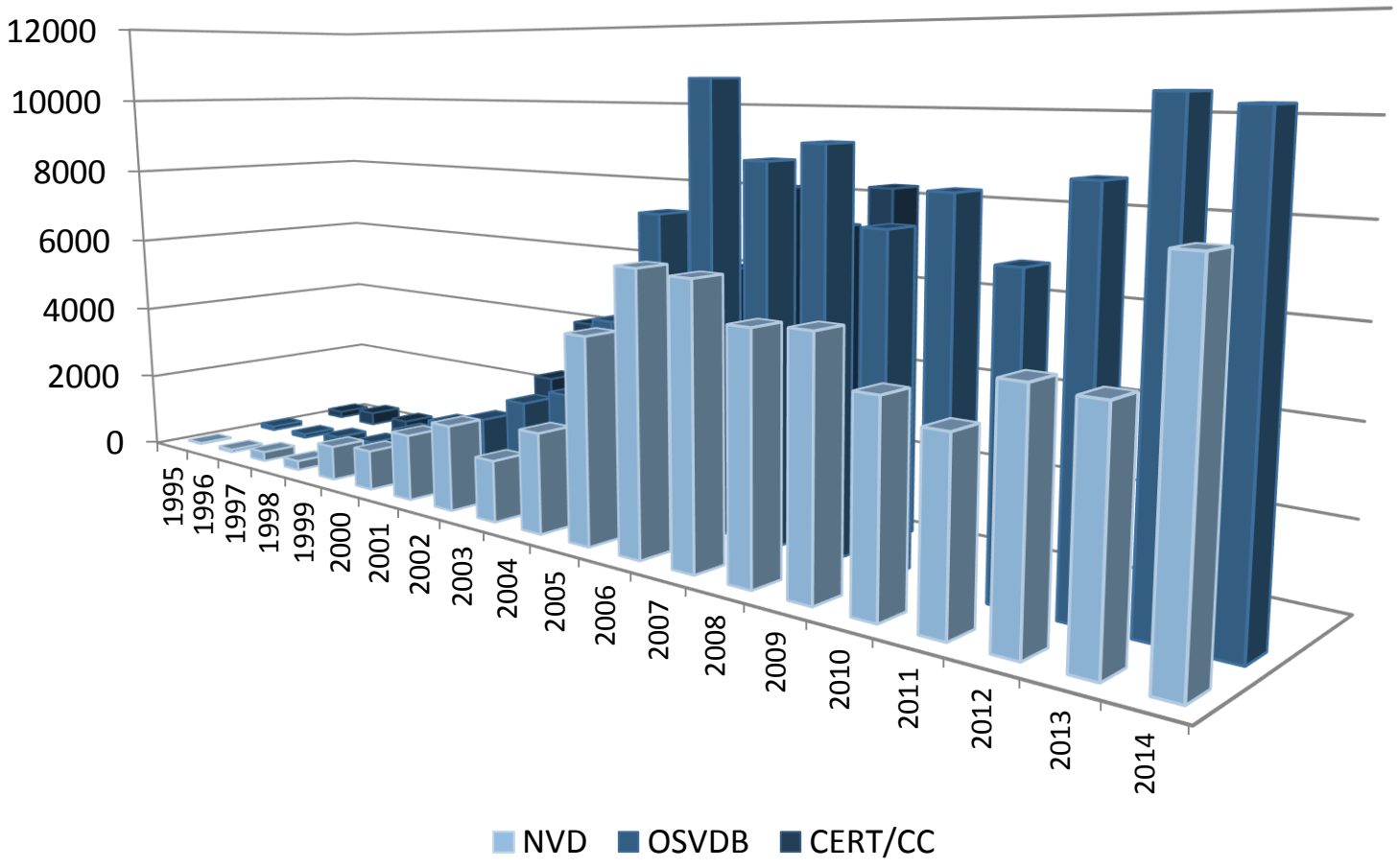
- Notable recent attacks/vulnerabilites:
  - HackingTeam 0-days
    - Several previously unknown vulnerabilities (Adobe Flash, Internet Explorer, Windows) complete with exploit code leaked.
  - Stagefright
    - Vulnerability in Android allows attacks by sending a malicious MMS
  - DoS vulnerability in BIND9 DNS server software
    - Runs on about 75% of the world’s DNS severs…

Common types of defects

- Buffer overflows
- Race conditions
- Encoding bugs
- Double free
- Integer overflows
- Memory leaks
- Format string bugs
- Cross-site scripting
- …

- There are lots of different kinds of defects!
- And those are only the kinds we know of…

How common are vulnerabilities



Software development today

- Developers are concerned with functionality, not with security
  - Security is often an afterthought and an add-on feature
  - Developers often don’t know a lot about security
  - Security principles are often not followed
- Customers don’t require security
  - Customers are often not aware of risks and threats
  - Security costs a lot but provides no direct benefits
- Software is **big and complex**

What can we do?

Secure software development

- Create **security awareness**
- Software development with **security in mind**
- Articulated **security requirements**
- Security in the **specification, architecture and design**
- Secure coding **guidelines and patterns**
- Independent **review and evaluation**

TDDC90 topics at a glance

- Create **security awareness**
- Software development with **security in mind**
  - ✓ Common vulnerabilities in programs written in C/C++, attack methods and mitigations
  - ✓ Web security: Common vulnerabilities and attacks
- Articulated **security requirements**
- Security in the **specification, architecture and design**
- Secure coding **guidelines and patterns**
  - ✓ Secure software development processes
  - ✓ Security requirements
  - ✓ Security modelling
- Independent **review and evaluation**
  - ✓ Code reviews
  - ✓ Static analysis
  - ✓ Software accreditation
  - ✓ Security testing

Organization of the course

Organization

- 10 lectures
  - One industry guest lecture
- 3 mandatory labs
  - Pong – the insecure ping
  - Static analysis
  - Web security
- Examination:
  - Written exam (3 hp)
  - Labs (3 hp)

Prerequisites

- Required:
  - Basic computer security course
  - Programming experience
  - Course in software engineering

- Recommended:
  - Operating systems and assembly programming basics
  - Some prior experience with C-programming
  - Basic course in logic
  - Basic web programming
    (HTML, JavaScript, some server-side language)

Lectures

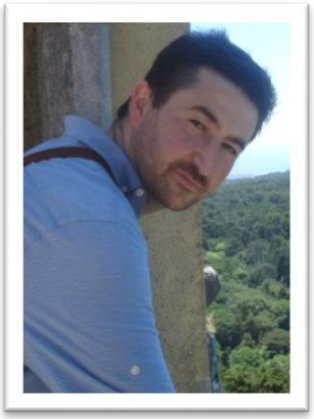
- Secure software development (1 lecture)
  Given by Marcus Bendtsen
  - Secure software development processes
  - Secure design patterns
  - Modeling and risk analysis

- Vulnerabilities and exploits (2 lectures)
  Given by Ulf Kargén
  - Common vulnerabilities in C/C++ programs
  - Known attack techniques
  - OS and compiler mitigations

Lectures (continued)

- Code reviews (1 lecture)
  Given by Kristian Sandahl
  - Software inspections and other techniques

- Static analysis (2 lectures)
  Given by Ahmed Rezine
  - Introduction to static analysis
    - Abstract interpretation
    - Symbolic execution



Lectures (continued)

- Web security (1 lecture)
  Given by Marcus Bendtsen
  - Common vulnerabilities in web applications
  - Attack techniques and protections

- Industry guest lecture
  Given by Susanne Frank, Combitech
  - Software security accreditation

Lectures (continued)

- Security testing and course wrap-up (1 lecture)
  Given by Ulf Kargén
  - Fuzzing, concolic testing
  - Course wrap-up

Labs

- Pong – the insecure ping
  - Perform a code review to find vulnerabilities
  - Exploit a buffer overflow to gain root
  - Fix all vulnerabilities
- Static
  - Study common static analysis techniques described in the lectures
- Websec
  - Deliberately vulnerable web app
  - Study common weaknesses and understand attack techniques

Labs

- Two groups for each lab
  - Different assistants for each lab – see lab page on course web
- Webreg signup deadline **11 November**
  - Unregistered students not allowed to sign up!
- Students are required to work in pairs
  - If you sign up alone, we may randomly group you with another student.
- **Hard** deadline for handing in solutions is **December 16th**
  - Complete all labs at least one week before this to allow time for corrections and re-submission
  - Hand in solutions continuously during the study period – don't save everything for the last week!
  - Start with labs as early as possible, especially Pong!

Reading material

- No course book (no one book covers all topics in the course)
- Mandatory reading:
  - Papers/articles, web resources, and lecture slides
  - Lectures don’t cover all articles, and vice versa
- Also a list of extra reading for interested students
  - Not needed for exam

Questions?